# $O$(log$^\ast n$) Time Parallel Algorithm for Computing Bounded Degree Subgraphs

AUTHOR(S):

Uchida, Tomoyuki

数理解析研究所講究録
第 754 巻 1991 年 104-114

104

# $O(\log^* n)$ Time Parallel Algorithm for Computing Bounded Degree Subgraphs

Tomoyuki Uchida[†]
*Department of Information Systems,*
*Kyushu University 39, Kasuga 816, Japan.*

31 January 1991

**Abstract**

We describe a fast algorithm that finds a maximal vertex-induced (resp., edge-induced) subgraph of degree at most $k$ for a graph with maximum degree $\Delta$ for $k \geq 0$ (resp., $k \geq 1$). These algorithms run in $O(\log^* n)$ time for constant degree graphs using the coloring algorithm given by Goldberg et al.

## 1 Introduction

Karp and Wigderson [KW85] and Luby [Lub85] have shown that the problem of finding a maximal independent set of a graph, called MIS, is in NC, which is known to be the class of problems computable by PRAMs with polynomial number of processors in $O((\log n)^k)$ time for some $k \geq 0$. However, Goldberg et al. [GPS87] and Jung and Mehlhorn [JM88] have shown that MIS for constant degree graphs can be computed in $O(\log^* n)$ time using $O(n)$ processors on an EREW PRAM by employing the deterministic coin tossing technique developed by Cole and Vishkin [CV86]. This technique is a method that finds a $\log n$-ruling set in $O(1)$ time using $O(n)$ processors under the assumption that each processor is capable of executing simple word and bit operations including unary-to-binary conversion and bit-wise boolean operations. By exploiting this technique, Goldberg et al. [GPS87] have shown that the coloring problem for constant degree graphs can be solved in $O(\log^* n)$ time on an EREW PRAM using $O(n)$ processors.

In this paper, we are interested in designing a fast algorithm that finds a maximal set of vertices (edges) which forms a subgraph of degree at most $k$ for a graph with

[†]Mailing address: Research Institute of Fundamental Information Science, Kyushu University 33, Fukuoka 812, Japan (e-mail: uchida@rifis.sci.kyushu-u.ac.jp).

maximum degree $\Delta$ by using the deterministic coin tossing technique. Shoudai and Miyano [SM90, SM91] have shown that the problem of finding a maximal subset of vertices (resp., edges) whose induced subgraph is of degree at most $k$, called VIMS($k$) (resp., EIMS($k$)) is in NC. They have shown that VIMS($k$) is in NC by using the NC algorithm for MIS devised by [KW85, Lub85] and EIMS($k$) is in NC by using the NC algorithm for the maximal matching problem. In this paper, however, we show that VIMS($k$) (resp., EIMS($k$)) for constant degree graphs can be computed in $O(\log^* n)$ time on an EREW PRAM using $O(n)$ processors by developing a method that employs the coloring algorithm devised by [GPS87].

In Section 2 we give some necessary definitions and the coloring algorithm given by [GPS87]. Then in Section 3 we show a fast algorithm for VIMS($k$) that uses the coloring algorithm. Section 4 gives a new algorithm for EIMS($k$) that also uses the coloring algorithm. In Section 5 we discuss the comparison between the algorithms shown in this paper and the algorithms presented by [SM90]. Our discussion yields that our algorithms find the solutions faster than the algorithms by [SM90] for the graphs with degree $\Delta = o(\log n)$ (we denote $f(n) = o(g(n))$ if $\limsup_{n\to\infty} f(n)/g(n) = 0$).

## 2 Preliminaries and Definitions

This section gives some definitions, graph notations and the computational model used throughout the paper.

We consider a graph $G = (V, E)$ as an undirected graph without any multiple edges and self-loops. Let $|V| = n$. For a subset $U \subseteq V$, we define $E[U] = \{\{u, v\} \in E \mid u, v \in U\}$. The graph $G[U] = (U, E[U])$ is called the *vertex-induced subgraph* of $U$. We define $V[F]$ to be the set of end points of the edges in $F$ for a subset $F \subseteq E$. We denote by $\langle F \rangle = (V[F], F)$ the graph formed from $F$ and call it the *edge-induced subgraph* of $F$. For a vertex $u$, the degree of $u$ is denoted by $deg_G(u)$. Maximum degree of $G$ is defined by $\Delta = \max\{deg_G(w) \mid w \in V\}$.

A coloring $C$ of $G$ is a mapping $C : V \to \mathbf{N}$ from the vertices to positive integers (colors), and it is *valid* if no two adjacent vertices have the same color.

**Definition 1.** Let $G = (V, E)$ be a graph with the maximum degree $\Delta$. The *vertex-coloring problem* is to find a valid coloring of $G$ that uses at most $\Delta + 1$ colors.

**Definition 2.** Let $G = (V, E)$ be a graph and let $k \geq 0$ be any integer. The *maximum degree $k$ vertex-induced subgraph problem* (VIMS($k$)) is to find a maximal subset $U \subseteq V$ such that $G[U]$ is of degree at most $k$.

**Definition 3.** Let $G = (V, E)$ be a graph and let $k \geq 1$ be any integer. The *maximum degree k edge-induced subgraph problem* (EIMS($k$)) is to find a maximal subset $F \subseteq E$ such that $\langle F \rangle$ is of degree at most $k$.

The following notation is used:

$$\begin{aligned} \log^{(0)} x &= x \\ \log^{(1)} x &= \lceil \log_2 x \rceil \\ \log^{(i)} x &= \lceil \log \log^{(i-1)} x \rceil \\ \log^* x &= \min\{i \mid \log^{(i)} x \leq 1\}. \end{aligned}$$

We assume a PRAM model of computation where each processor is capable of executing simple word and bit operations. The word length is assumed to be $O(\log n)$. The word operations include bit-wise boolean operations, integer comparisons and unary-to-binary conversion. We use exclusive-read exclusive-write (EREW) PRAM and concurrent-read concurrent-write (CRCW) PRAM as appropriate.

Goldberg, Plotkin and Shannon [GPS87] have presented a coloring algorithm that yields the following theorem:

**Theorem 1 (Goldberg et al. [GPS87]).** *Given a graph $G = (V, E)$ with the maximum degree $\Delta$, a valid coloring of $G$ with $\Delta+1$ colors can be computed in $O(\Delta \log \Delta \times (\Delta + \log^* n))$ time on an EREW PRAM using $O(\Delta n)$ processors.*

Moreover, they have also presented the following theorem for MIS by using the coloring algorithm:

**Theorem 2 (Goldberg et al. [GPS87]).** *An MIS in a graph $G = (V, E)$ with the maximum degree $\Delta$ can be computed in $O(\Delta \log \Delta \times (\Delta + \log^* n))$ time on an EREW PRAM using $O(\Delta n)$ processors.*

# 3 Finding Bounded Degree Vertex-Induced Maximal Subgraphs

In this section we describe an algorithm which computes VIMS($k$) in a graph whose degree is at most $\Delta$.

Our VIMS-algorithm is shown in Figure 1. Let $k$ be a positive integer. The VIMS-algorithm takes a graph $G = (V, E)$ with the maximum degree $\Delta$ as input, and outputs a maximal subset $S \subseteq V$ such that $G[S]$ is of degree at most $k$.

For each $i = 0, \ldots, \Delta$, let $C_i(V) = \{v \in V \mid C(v) = i\}$. For a subset $S \subseteq V$ and a vertex $v \in V$, let $U_v[S]$ be the set of vertices in $S$ that are adjacent to $v$. For subsets $W$ and $U$ of vertices with $W \cap U = \emptyset$, let $E_U^W = \{\{v, w\} \mid$ there is $u \in U$ with $w \neq v$ such that $v, w \in W$ and $\{v, u\} \in E$, $\{w, u\} \in E$ $\}$.

Initially let $S = \emptyset$. First, the algorithm colors a graph $G$ with $\Delta + 1$ colors. Next, for each $i = 0, \ldots, \Delta$, the algorithm decides which vertices colored $i$ are added to $S$.

Formally the algorithm is described as follows:

The algorithm consists of the two phases. In the first phase (1), it colors all vertices in $V$ with $\Delta + 1$ colors by using the coloring algorithm by [GPS87]. The phase (1) guarantees that we can simultaneously deal with every vertex by taking notice of its color. The second phase (2) consists of the five parts. The phase (2) is iterated $\Delta + 1$ times. In the first part (i), we can add a vertex $v \in C_i(V)$ to $S$ if $Deg(v) \leq k$, where $Deg(v) = deg_{G[S \cup \{v\}]}(v)$. We note that, after executing the part (i), there may exist a vertex $u \in S$ with $Deg(u) > k$. Therefore, we must reduce the degree of a vertex $u$ until $Deg(u) \leq k$ in the following part: In the second part (ii) we add $v$ to $V'$ for a vertex $v$ in $S$ with $Deg(v) > k$. In the third part (iii) we add $u$ to $V''$ for a vertex $u$ in $S \cap C_i(V)$ such that there exists a vertex $w$ in $U_u[S]$ with $Deg(w) > k$. Since the degree of the induced subgraph $G[S]$ must be at most $k$, we delete from $S$ the vertices in $V''$ in the fourth part (iv). Therefore, after executing the part (iv), the induced subgraph $G[S]$ is of degree at most $k$ but may not be maximal. Hence, the fifth part (v) constructs the maximal induced subgraph $G[S]$ which is of degree at most $k$ by using the coloring $C'$ of the graph $(V'', E_{V''}^{V''})$. By the definitions of the $V''$ and $E_{V''}^{V''}$, it is easy to see that the degree of the graph $(V'', E_{V''}^{V''})$ is at most $k \times \Delta$. By using the coloring $C'$, we can make the induced subgraph $G[S]$ maximal. Therefore, the coloring $C'$ assigns a positive integer in $\{0, \ldots, k \times \Delta\}$ to each vertex in $V'$.

By the VIMS-algorithm, we can prove the following theorem:

**Theorem 3.** *Let $k$ $(0 \leq k < \Delta)$ be a positive integer. Given a graph $G = (V, E)$ of degree at most $\Delta$, VIMS(k) can be computed in $O(k\Delta^2 \log \Delta \times (k\Delta + \log^* n))$ time on an EREW PRAM using $O(k\Delta n)$ processors.*

**Proof.** We show the correctness of the algorithm. For each $i = 0, \ldots, \Delta$, let $S_i$ be the contents of $S$ at the end of $i$th iteration in the phase (2). We assume that the maximal induced subgraph $G[S_{i-1}]$ is of degree at most $k$.

We show that the induced subgraph $G[S_i]$ in $G[S_{i-1} \cup C_i]$ is of degree at most $k$ and maximal. Clearly, after executing the part (iv), the graph $G[S]$ is of degree at most

**VIMS-Algorithm:**

$S \leftarrow \emptyset$;
(1) Computes the *vertex-coloring* $C$ of $(V, E)$;
(2) **for** $i \leftarrow 0$ **to** $\Delta$ **do**
(i) $S \leftarrow S \cup \{v \in C_i(V) \mid Deg(v) < k\}$;
(ii) $V' \leftarrow \{v \in S \mid Deg(v) > k\}$;
(iii) $V'' \leftarrow \{v \in S \cap C_i(V) \mid \exists w \in U_v[S]$ with $Deg(w) > k\}$;
(iv) $S \leftarrow S - V''$;
(v) **if** $V' \neq \emptyset$ **then**
Computes the *vertex-coloring* $C'$ of $(V'', E^{V''}_{V'})$;
**for** $j \leftarrow 0$ **to** $k\Delta$ **do**
$S \leftarrow S \cap \{v \in C'_j(V'') \mid Deg(v) \leq k \wedge (\nexists w \in U_v[S]$ with $Deg(w) \geq k)\}$
**od**
**od**

Figure 1: The algorithm for VIMS($k$)

$k$, but may not be maximal. Hence, we pay attention to the fifth part (v) and show that the induced subgraph $G[S_i]$ is of degree at most $k$ and is maximal. For any two vertices $v, w$ in $S$ which are adjacent to a vertex in $V'$, we can see that $C'(v) \neq C'(w)$, since an edge $\{v, w\}$ is in $E^{V''}_{V'}$ by the definitions of $V', V''$ and $E^{V''}_{V'}$. Therefore, by executing the part (v), the induced subgraph $G[S]$ can be made maximal, keeping the condition that the degree of the graph $G[S]$ is at most $k$. Hence, we can see that the induced subgraph $G[S_i]$ is of degree at most $k$ and is maximal.

Finally, we show that the algorithm runs in $O(k\Delta^2 \log \Delta \times (k\Delta + \log^* n))$ time using $O(k\Delta n)$ processors. The phase (1) runs in $O(\Delta \log \Delta \times (\Delta + \log^* n))$ time using $O(\Delta n)$ processors by Theorem 1. Next we show the complexity of the phase (2). The part (i) takes $O(1)$ time using $n$ processors. Since the degree of the graph $G$ is at most $\Delta$, the part (ii), the part (iii) and the part (iv) run in $O(\Delta)$ time using $O(\Delta n)$ processors. The time of the part (v) is bounded by the time to color the graph $(V'', E^{V''}_{V'})$. That is, the part (v) takes $O(k\Delta \log \Delta \times (k\Delta + \log^* |V''|))$ time. Hence, the phase (2) takes $O(k\Delta \log \Delta \times (k\Delta + \log^* |V''|))$ time using $O(k\Delta n)$ processors. Therefore, the algorithm runs in $O(k\Delta^2 \log \Delta \times (k\Delta + \log^* n))$ time using $O(k\Delta n)$ processors. □

When a constant degree graph is given as an input, the following corollary can be immediately proved by Theorem 3:

**Corollary 1.** *Let $k$ be a positive integer. Given a constant degree graph, VIMS($k$) can be computed in $O(\log^* n)$ time on an EREW PRAM using a linear number of processors.*

# 4 Finding Bounded Degree Edge-Induced Maximal Subgraph

This section gives an algorithm that computes EIMS($k$) for graphs with the maximum degree $\Delta$. The EIMS-algorithm is shown in Figure 2.

Let $k$ ($1 \leq k \leq \Delta$) be a positive integer. The algorithm takes a graph $G = (V, E)$ with the maximum degree $\Delta$ as input, and outputs a maximal subset $F \subseteq E$ such that $\langle F \rangle$ is a graph of degree at most $k$.

For a subset $V' \subseteq V$, let $E_{i,j}[V'] = \{\{v, w\} \in E[V'] \mid C(v) = i \text{ and } C(w) = j\}$ for each $0 \leq i < j \leq \Delta$. Let $W$ be an independent set of $G = (V, E)$ and $X \subseteq E$ be a set of edges such that $W \subseteq V[X]$. Then let $H[W, X]$ be a minimal subset of $X$ such that $W \subseteq V[H[W, X]]$. Since $W$ is an independent set, such $H[W, X]$ can be easily computed by selecting an edge $\{v, w\} \in X$ for each $v \in W$.

Formally the algorithm is described as follows:

Initially let $F = \emptyset$ and $V' = V$. The algorithm consists of the two phases. First the algorithm colors an input graph $G$ with $\Delta + 1$ colors. This is executed in the first phase (1). The algorithm colors all vertices with $\Delta + 1$ colors. The phase (1) guarantees that we can simultaneously deal with every edge with respect to color.

Next the algorithm decides whether to add to $F$ a edge $\{v, w\}$ with $C(v) = i$ and $C(w) = j$ at each iteration $(i, j)$ for each $0 \leq i < j \leq \Delta$. The parts (i)-(iv) are repeated until there exists no vertex $v$ in $C_i(V')$ such that $deg_{\langle F \rangle}(v) < k$ and there exists an edge $\{v, w\} \in E_{i,j}[V']$. In the first part (i), the algorithm adds to $E'$ an edge $\{v, w\}$ which is selected from $E_{i,j}[V']$ for each vertex $v$ in $V'$. The algorithm adds the vertices in $E'$ to $F$. Now we remark that, after executing the part (i), there may exist a vertex $v$ with $deg_{\langle F \rangle}(v) > k$. Hence, in the second part (ii) the algorithm deletes every edge $\{v, w\}$ in $E'$ from $F$ such that $v \in C_i(V')$, $w \in C_j(V')$ and $deg_{\langle F \rangle}(w) > k$. For each vertex $v$ in $U = \{s \in C_j(V') \mid deg_{\langle F \rangle}(s) > k\}$, the third part (iii) adds to $F$ an edge incident to $v$ and deletes $v$ from $U$. The part (iii) is repeated until $U = \emptyset$. In the fourth part (iv), the algorithm deletes from $F$ every vertex $v$ with $deg_{\langle F \rangle}(v) = k$, and deletes from $E_{i,j}[V']$ the edges in $E'$.

**EIMS-Algorithm:**

$F \leftarrow \emptyset$; $V' \leftarrow V$;
(1) Computes the *vertex-coloring* $C$ of $G = (V, E)$;
(2) **for** $i \leftarrow 0$ **to** $\Delta$ **do**
  **for** $j \leftarrow i+1$ **to** $\Delta$ **do**
    $I \leftarrow E_{i,j}[V']$;
    **while** $\{v \in C_i(V') \mid deg_{\langle F \rangle}(v) < k \wedge (\exists \{v, w\} \in I \text{ with } w \in C_j(V'))\} \neq \emptyset$ **do**
(i)  $E' \leftarrow H[C_i(V'), I]$;
      $F \leftarrow F \cup E'$;
(ii)  $U \leftarrow \{s \in C_j(V') \mid deg_{\langle F \rangle}(s) > k\}$;
      $F \leftarrow F - \{\{v, w\} \in E' \mid v \in C_i(V') \text{ and } w \in U\}$;
(iii)  **while** $U \neq \emptyset$ **do**
        $F \leftarrow F \cup H[U, E']$;
        $U \leftarrow U - \{t \in U \mid deg_{\langle F \rangle}(t) = k\}$
      **od**
(iv)  $V' \leftarrow V' - \{v \in V' \mid deg_{\langle F \rangle}(v) = k\}$;
      $I \leftarrow E_{i,j}[V'] - E'$;
      $E' \leftarrow \emptyset$
    **od**
  **od**
**od**

Figure 2: The algorithm for EIMS($k$)

By the EIMS-algorithm, we can prove the following theorem:

**Theorem 4.** *Let $k$ $(1 \leq k < \Delta)$ be a positive integer. Given a graph $G = (V, E)$ with the maximum degree $\Delta$, EIMS($k$) can be computed in $O(\max(\Delta \log \Delta \times (\Delta + \log^* n), \Delta^3 \log \Delta))$ time on an EREW PRAM using $O(\Delta n)$ processors.*

**Proof.** We show the correctness of the algorithm. For $0 \leq i < j \leq \Delta$, let $F_{i,j}$ be the contents of $F$ at the end of $(i, j)$th iteration in the phase (2). We assume that the maximal induced subgraph $\langle F_{i,j} \rangle$ is of degree at most $k$.

Let $\{v, w\}$ be an edge with $C(v) = i$ and $C(w) = j + 1$. The edge $\{v, w\}$ is added to $F$ if $deg_{\langle F \rangle}(v) < k$. Here, we remark that the degree of $v$ increases only by at most one in every iteration, but we do not know how the degree of $w$ increases. However, the edge $\{v, w\} \in E'$ with $deg_{\langle F \rangle}(w) > k$ is deleted from $F$ in the part (ii). Therefore,

the algorithm can make the degree of $v$ on $\langle F_{i,j+1} \rangle$ at most $k$ in the part (i), and make the degree of $w$ on $\langle F_{i,j+1} \rangle$ at most $k$ in the part (ii). Hence, we can see that, after executing the part (ii), the induced subgraph $\langle F_{i,j+1} \rangle$ is of degree at most $k$, but it may not be maximal. It is easy to see that, after executing the part (iii), $\langle F_{i,j+1} \rangle$ is maximal. Since the algorithm deals with all edges in $E_{i,j+1}[V']$ in the parts (i)-(iv), it is straightforward to see that the induced subgraph $\langle F_{i,j+1} \rangle$ is maximal.

Similarly, we assume that the maximal induced subgraph $\langle F_{i,\Delta} \rangle$ is of degree at most $k$. We can see that the induced subgraph $\langle F_{i+1,i+2} \rangle$ is of degree at most $k$ and is maximal. Hence,

$$F_{1,2} \subseteq \cdots \subseteq F_{i,j} \subseteq F_{i,j+1} \subseteq \cdots \subseteq F_{i,\Delta} \subseteq F_{i+1,i+2} \subseteq \cdots \subseteq F_{\Delta-1,\Delta}.$$

Therefore, the algorithm can compute EIMS($k$).

Next, we show that the algorithm runs in $O(\max(\Delta \log \Delta \times (\Delta + \log^* n), \Delta^3 \log \Delta))$ time using $O(\Delta n)$ processors on an EREW PRAM. In the phase (1), the algorithm takes $O(\Delta \log \Delta \times (\Delta + \log^* n))$ time using $O(\Delta n)$ processors on an EREW PRAM. In the phase (2), It takes $O(\Delta^3 \log \Delta)$ time using $O(\Delta n)$ processors on an EREW PRAM. Therefore, we can see that the algorithm runs in $O(\max(\Delta \log \Delta \times (\Delta + \log^* n), \Delta^3 \log \Delta))$ time using $O(\Delta n)$ processors on an EREW PRAM. □

When a constant degree graph is given as an input, the following corollary holds:

**Corollary 2.** *Let $k$ ($1 \leq k$) be a positive integer. Given a constant degree graph, EIMS($k$) can be computed in $O(\log^* n)$ time on an EREW PRAM using a linear number of processors.*

# 5 Discussion

The coloring of a constant degree graph allows us to construct fast algorithms for VIMS($k$) and EIMS($k$). As a consequence, for VIMS($k$) and EIMS($k$) with a constant degree graph as an input, we have the fast algorithms that run in $O(\log^* n)$ time using a linear number of processors by exploiting the coloring algorithm presented by Goldberg et al. [GPS87].

Shoudai and Miyano [SM90] have presented the algorithms for VIMS($k$) and EIMS($k$). First, we look at their VIMS algorithm which computes VIMS($k$) for a integer $k \geq 0$. Their VIMS algorithm iterates the following operations $k^2$ times, where initially let $W = V$ and $U = \emptyset$:

- Finding a maximal independent set $I$ of $H_U^W = (W, E[W] \cup E_U^W)$.

- Adding $I$ to $U$ while vertices which make the degree of some vertex greater than $k$ are deleted from $W$ together with $I$.

It is easy to see that the degree of the graph $H_U^W = (W, E[W] \cup E_U^W)$ increases by at most $\Delta^2$ and the running time of their VIMS algorithm is bounded by the time of the MIS algorithm. For example, by using the MIS algorithm by Luby [Lub85], their VIMS algorithm runs in $O((\log n)^2)$ time on an EREW PRAM using $O(n^2 m)$ processors, where $|V| = n$ and $|E| = m$. By using the MIS algorithm of Theorem 2, for a graph with the maximal degree $\Delta$, their VIMS algorithm runs in $O(k^2\Delta^2 \log \Delta(\Delta^2 + \log^* n))$ time on an EREW PRAM using $O(\Delta^2 n)$ processors. We remark that, since the degree of the graph $H_U^W = (W, E[W] \cup E_U^W)$ increases by at most $\Delta^2$, their VIMS algorithm needs at most $\Delta^2$ colors to color its graph $H_U^W$. Hence, using the MIS algorithm of Theorem 2 enables us to construct a very fast VIMS algorithm using $O(\Delta n)$ processors for the graph which the maximal degree $\Delta$ is $o(\log n)$.

On the other hand, the VIMS-algorithm that computes VIMS($k$) for $k \geq 0$ uses the coloring algorithm of Theorem 1 and the degree of the graph $(V'', E_{V''}^{V''})$ increases only by at most $k\Delta$ for a graph with the maximal degree $\Delta$. Therefore, the VIMS-algorithm runs in $O(k\Delta^2 \log \Delta(k\Delta + \log^* n))$ time on an EREW PRAM using $O(k\Delta n)$ processors. As a result, if the maximum degree of an input graph is less than $O(\log n)$, then the VIMS-algorithm get the solutions to VIMS($k$) faster than the VIMS algorithm by [SM90]. Otherwise their VIMS algorithm runs faster than the VIMS-algorithm.

Next, we look at the EIMS algorithm [SM90] that computes EIMS($k$) for an integer $k > 0$. Their EIMS algorithm repeats the following operations $2k$ times, where initially $Z = E$ and $F = \emptyset$:

- Finding a maximal matching $M$ of $\langle Z \rangle$.
- Adding $M$ to $F$ while edges which make the degree of some vertex greater than $k$ are deleted from $Z$ together with $M$.

It is easy to see that the running time of their EIMS algorithm is bounded by the time of the algorithm for the maximal matching problem. For example, by using the maximal matching algorithm devised by [IS86], for a graph $G = (V, E)$, their EIMS algorithm runs in $O((\log m)^3)$ time using $n + m$ processors, where $|V| = n$ and $|E| = m$. Goldberg et al. [GPS87] have shown that the maximal matching problem for planar graphs in $O(\log n \log^* n)$ time on a CRCW PRAM using a linear number of processors. By using this result, their EIMS algorithm for planar graphs runs in $O(\log n \log^* n)$ time using a linear number of processors.

On the other hand, the EIMS-algorithm runs in $O(\log^* n)$ time using a linear number of processors for constant degree graphs by Corollary 2.

Hence, we can see that for graphs whose maximal degree is $o(\log n)$, the EIMS-algorithm runs faster than the EIMS algorithm by [SM90], but for graphs with maximal degree $\Delta \neq o(\log n)$, their EIMS algorithm runs faster than the EIMS-algorithm.

Finally it is easy to see that, for a planar graph, EIMS($k$) can be computed in $O(\log n \log^* n)$ time on a CRCW PRAM, and in $O(\log n(\log \Delta + \log^* n))$ time on an EREW PRAM, using a linear number of processors by applying the result of the following theorem by [GPS87] to the EIMS-algorithm:

**Theorem 5 (Goldberg et al. [GPS87]).** *Given a planar graph, a valid coloring with 5 colors can be computed using $n$ processors and $O(\log n \log^* n)$ time on a CRCW PRAM, and $O(\log n(\log \Delta + \log^* n))$ time on an EREW PRAM.*

On the other hand, since the VIMS-algorithm colors the graph $(V'', E_{V''}^{V''})$ which is not a planar graph in the part (v) and since the VIMS algorithm by [SM90] finds the maximal independent set of the non-planar graph $H_U^W = (W, E[W] \cup E_U^W)$, this theorem can be applied to neither of the algorithms.

# References


[CV86] R. Cole and U. Vishkin. Deterministic coin tossing with applications to optimal parallel list ranking. *Inform. Control*, Vol. 70, No. 1, pp. 32–53, July 1986.

[GPS87] A. V. Goldberg, S. A. Plotkin, and G. E. Shannon. Parallel symmetry-breaking in sparse graphs. In *Proc. 19th ACM STOC*, pp. 315–324, May 1987.

[IS86] A. Israeli and Y. Shiloach. An improved parallel algorithm for maximal matching. *Inform. Process. Lett.*, pp. 57–60, Jan. 1986.

[JM88] H. Jung and K. Mehlhorn. Parallel algorithms for computing maximal independent sets in trees and for updating minimum spanning trees. *Inform. Process. Lett.*, Vol. 27, No. 5, pp. 227–236, 1988.

[KW85] R. M. Karp and A. Wigderson. A fast parallel algorithm for the maximal independent set problem. *J. Assoc. Comput. Mach.*, Vol. 32, pp. 762–773, 1985.

[Lub85] M. Luby. A simple parallel algorithm for the maximal independent set problem. In *Proc. 17th ACM STOC*, pp. 1–10, May 1985.

[SM90] T. Shoudai and S. Miyano. Bounded degree maximal subgraph problems are in NC. In *Proc. Toyohashi Symposium on Theoretical Computer Science*, pp. 97–101, 1990.

[SM91] T. Shoudai and S. Miyano. Using maximal independent sets to solve problems in parallel. RIFIS-TR-CS-36, Research Institute of Fundamental Information Science, Kyusyu University, 1991.